Выводы. Таким образом, в статье был рассмотрен программный продукт FxCop, позволяющий автоматизировать процесс рецензирования исходного кода путём поиска известных ошибок. Была показана необходимость разработки собственных правил, предъявляемых к исходному коду, указаны сложности существующего решения. Предложено решение на основе использования онтологии правил, позволяющее упростить разработку собственных правил, приведены его преимущества. Намечены дальнейшие направления развития работы.

Список литературы: **1.** *A.Stellman, J.Greene.* Applied Software Project Management, O'Reilly, 2005. **2.** IEEE Std 1028-1997, IEEE Standard for Software Reviews. **3.** *И.Соммервил.* Инженерия программного обеспечения, 6-е издание. М. Вильямс, 2002. **4.** *V.R.Basili, R.W.Selby.* Comparing the effectiveness of software testing strategies, 1987. SE-13(12). **5.** *T.Gilb, D.Graham.* Software Inspection. Addison-Wesley, 1993. **6.** *M.E.Fagan* Advances in software inspections. – IEEE Trans. on Software Engineering, 1986, SE-12(7). **7.** *J.Poley* Best Practices: Code Reviews, Microsoft Corporation, October 2007, http://msdn2.microsoft.com/en-us/library/bb871031.aspx. **8.** *J.D. Meier et al.*, How To: Perform a Security Code Review for Managed Code (Baseline Activity), Microsoft Corporation, October 2005, http://msdn2.microsoft.com/en-us/library/ms998364.aspx. **9.** FxCop Home Page, http://msdn2.microsoft.com/en-us/library/bb429476(VS.80).aspx. **10.** *K.Cwalina, B.Abrams.* Framework Design Guidelines: Conventions, Idioms, and Patterns for Reusable .NET Libraries, Addison-Wesley Professional, 2005. **11.** *R.Neches, R.E.Fikes et al.* Enabling technology for knowledge sharing. AI Magazine 12(3), 1991, p.36–56. **12.** *T.R.Gruber*. A translation approach to portable ontology specification. Knowledge Acquisition 5(2), 1993, p.199–220. **13.** *R.Studer, V.R.Benjamins, D.Fensel.* Knowledge Engineering: Principles and Methods. IEEE Transactions on Data and Knowledge Engineering, Vol.25(1-2), 1998, p.161–197. **14.** *A.Gómez-Pérez, M.Fernández-López, O.Corcho.* Ontological Engineering, Springer-Verlag, London 2004. **15.** *Т.А. Гаврилова, В.Ф. Хорошевский.* Базы знаний интеллектуальных систем, СПб.:Питер, 2000. **16.** *C. Calero , F. Ruiz, M. Piattini.* Ontologies for Software Engineering and Software Technology, Springer-Verlag, Berlin Heidelberg 2006.

Поступила в редколлегию 20.02.08

УДК 004.728.8

К.І. ЯЦЕНКО, аспірант факультету кібернетики Київського Національного Університету імені Тараса Шевченка, kyatsenko@gmail.com

ПРОТОКОЛИ ДЛЯ ІНТЕГРАЦІЇ МОБІЛЬНИХ ПРИСТРОЇВ ІЗ СТАНЦІЯМИ ПО НАДАННЮ ПОСЛУГ ЧЕРЕЗ БЕЗДРОТОВІ ЕЛЕКТРОННІ КАНАЛИ

Запропоновано новий метод в області інформаційних технологій по наданню локальних послуг через безшовну інтеграцію мобільних пристроїв. Підхід базується на класі протоколів XDEP (XML data exchange protocol). Основним завданням цього підходу є заміна існуючих методів надання інформації, які функціонують завдяки спеціалізованим пристроям. Пропонується перенести цю функціональність на пристрої масового використання - персональні електронні записники, мобільні телефони, MP3 плеєри, та інші.

A new approach in the area of information technologies of providing local services via seamless integration of mobile devices has been proposed. The approach resides on the basis of XDEP (XML data exchange protocol). The solution is intended to substitute existing methods of information services providing, which reside on a specialized hardware. The functionality is to migrate to the commonly used devices like PDAs, Mobile Phones, MP3 players, etc.

1. Вступ. В сучасних динамічних умовах розвитку інформаційних технологій, рівень інтеграції мобільних пристроїв, персональних комп'ютерів та серверів через електронні канал є високим. Кожний мобільний пристрій містить персональну інформацію, яку користувачі потребують для вирішення атомарних питань. Персональні комп'ютери акумулюють інформацію, яку користувачі збирають на протязі деякого часу. Серверні системи представляють засіб довготривалого зберігання інформації від різних користувачів. Таким чином, інформацію, яку користувачі потребують в різні періоди часу можна класифікувати на три категорії:

Використання для дії (зберігається на мобільних пристроях)

Використання для діяльності (набір дій, зберігається на персональному комп'ютері)

Використання для історії та планування (набір різних діяльностей, які зберігаються на серверній платформі)

За цією класифікацію, дані зберігаються та організовуються на різних фізичних платформах. Такий підхід до зберігання інформації потребує постійної синхронізації обладнання. Логічний процес взаємодії різних програмно-апаратних комплексів розглядається в цій статті, яка пропонує уніфікований метод його організації.

Клас протоколів XDEP (XML Data Exchange Protocol) вперше представлений під час конференції ICTA 2007 [1]. Головною ідеєю цього класу

є представлення даних в універсальному форматі, який системи, за різними логічною та фізичною будовами, могли б інтерпретувати швидко та безпомилково. Підхід до побудови структури цього класу протоколів був схожий до структури веб-сервісів [3] розроблених W3C. Базою для веб-сервісів є мета опис даних, який легко інтерпретується будь-якою програмою [4].

Для того, щоб висвітлити наступний етап еволюції XDEP розглянемо наступний приклад. В Парижі, Франції, як альтернативу використання транспорту на бензині, комунальні служби запропонували використання велосипедів (послуга Velolib) за дуже символічну платню. Передбачається, що така послуга розвантажить автомобільний трафік та позитивно вплине на навколишнє середовище. Однак, з метою попередження можливості не цільового використання або крадіжки велосипедів, мандрівник має валідувати кредитну картку, перед тим, як скористатися новою послугою громадського транспорту. Велосипед можна взяти безкоштовно на півгодини, після чого, за кожні півгодини починаючи з одного євро, ціна прогресує в арифметичній прогресії. Платформи (місця, де можна отримати та здати велосипед) знаходяться по всьому місту, отже зміна велосипеду не представляє складнощів, та може бути виконана дуже швидко. Проте, мандрівники змушені кожного разу, під час заміни велосипеда повторювати операцію з введенням персональних даних. Такий процес може займати для однієї людьми близько 3 хвилини, що складає 10% від потенційного часу використання послуги. Враховуючи те, що завжди кількість автоматичних пунктів (екрани з клавіатурою) на платформі не перевищує 1, в разі черги, економія часу при використання таких послуг стає дуже сумнівною. На поточний момент єдиною можливістю вирішення цієї проблеми є встановлення додаткових автоматичних пунктів (хоча, в цьому випадку виникають складнощі з синхронізацією дій мандрівників, які хочуть взяти один і той самий велосипед).

В той час, в деяких країнах (таких, наприклад, як Україна) кількість користувачів мобільних телефонів перевищила 100% популяції. Це означає, що кожна людина має як мінімум один мобільний телефон. Як результат кожний користувач має постійно із собою персональну консоль із можливостями вводу та виводу даних. Інтерфейс для представлення даних для використання типу дії (або представлення атомарних послуг), може бути перенесений на будь-який персональний пристрій. Цей підхід відображає наступний крок еволюції XDEP.

2. Структура XDEP. Рис. 1 структурно відображає кон'юнктуру використання протоколів. Кожний протокол передбачений для дії в ролі медіатора

між різними клієнтськими та серверними платформами. Завдяки текстовому формату, протокол може бути легко адаптований під різні системи, що дозволяє його широке використання.

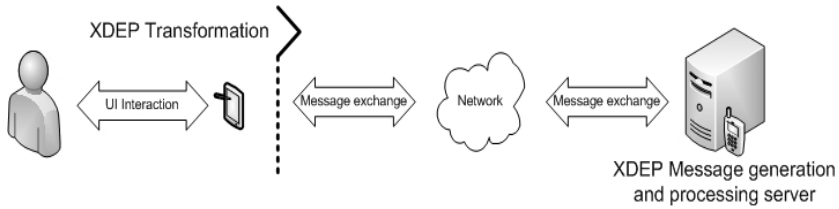

Рис. 1. Кон‘юнктура використання протоколів

Для конкурсу Microsoft Imagine Cup 2007 автор удосконалив XDEP для того, щоб надати можливість людям із ускладненнями зору отримувати різні інформаційні послуги використовуючи звичайні портативні пристрої. Інформація із серверної програми подавалась на мобільний пристрій з голосовими можливостями в форматі XDEP, яка представлялась користувачеві через голосовий інтерфейс. Архітектуру цієї системи представлено на Рис.2

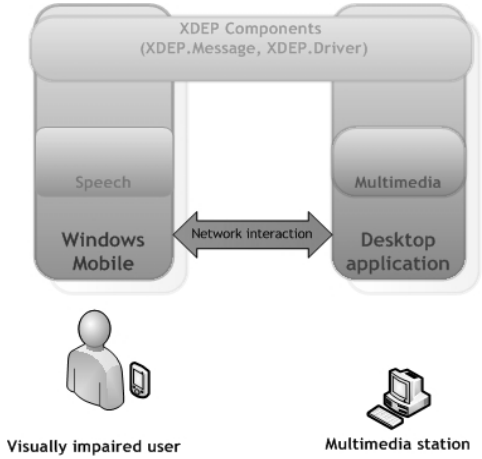

Рис. 2. Архітектура системи допомоги особам із ускладненнями зору

Нижче представлено приклад одніє із команд протоколу XDEP для проекту Fenestra :

```
<?xml version="1.0" encoding="utf-8"?>
<XDEPDriver ProtocolName =”Fenestra” ServerName=”Fenestra Server” start-
command=””>
 <Commands>
  <Command>
   <Name>protocol syntax</Name>
   <IsStatic>false</IsStatic>
   <IsDriverDescription>true</IsDriverDescription>
   <IsSilent>false</IsSilent>
   <Description>gets the protocol description</Description>
   <Parameters>
    < param >
     <ParamName>Protocol</ParamName>
     <PredecessorParamReference />
     <ParamType>String</ParamType>
     <ParamDefaultValue = “” />
     <Direction>Out</Direction>
    </ param >
   </Parameters>
   <Successors>
   </Successors>
  </Command>
  <Command>
   <Name>music play</Name>
   <IsStatic>true</IsStatic>
   <IsSilent>false</IsSilent>
   <Description>If you want to listen to some music</Description>
   <Parameters>
   </Parameters>
  </Command>
  <Command>
   <Name>music stop</Name>
   <IsStatic>true</IsStatic>
      <IsDriverDescription>false</IsDriverDescription>
      <IsSilent>false</IsSilent>
   <Description>If you want to stop listening to the music</Description>
   <Parameters>
   </Parameters>
  </Command>
 </Commands>
</XDEPDriver>
```

Цей приклад демонструє три команди з самоописуємою структурою. Одна з цих команд protocol syntax – базова команда для всіх протоколів побудованих із-за допомогою правил XDEP. Дві інші (music play команда, яка

запускає програвач музики та music stop, яка зупиняє аудіотрек) - специфічні команди даного прикладу.

При розгляді XDEP необхідно приділити додаткову увагу іменуванню команд, їх опису та параметрам. Може здатись, що досить великий опис команд збільшує їх розмір, таким чином збільшується і мережевий трафік. Однак, враховуючи швидкість росту потужності комп'ютерних мереж, цей факт можна ігнорувати. На противагу, такий підхід дозволяє інтегрувати різні прилади, навіть ті, які орієнтовані для людей з проблемами зору. Як результат клієнтські програми можуть успішно використовувати аудіо інтерфейси, що дозволяє не змінювати представлення даних (мета дані) та адаптувати команди для приладів з різним інтерфейсом.

Наступним кроком вивчення протоколу буде розгляд опису команд в загальному випадку.

```
<?xml version="1.0" encoding="UTF-8"?>
<XDEPDriver ProtocolName="TheNameOfTheCurrentProtocol"
ServerName ="TheNameOfTheServerDevice"
    startcommand="NameofTheStartingcommand">
<Command name="CommandName" isStatic="boolean"
description="userfriendlydescription">
<parameters>
    <param
            paramName="paramName"
            predecessorParamReference="paramName"
            paramType= "integer|boolean|string|list"
            maximumValues =”integer value”
 minValues=”integer value”
 validationRule=”regular expression”
            paramDefaultValue= "thevalue"
            direction="in|out|inout">
                <values>
                        <value key="value" value="value"/>
 </values>
    </param>
</parameters>
<successors>
<successor commandName="CommandName"
        autocall="bool"/>
 </successors>
    </Command>
```

Комунікація між клієнтськими та серверними програмами виконується, в командному режимі. Нижче пропонується загальний синтаксис запитів та відповідей протоколу: Запит виглядає наступним чином:

```
<XDEPInlineCommand commandName="CommandName">
  <parameters>
    <param paramName="paramName" paramValue="thevalue"/>
  </parameters>
</XDEPInlineCommand>
```

Відповідь:

```
<!--Response sent to the client's command-->
<XDEPInlineResponse commandName="commandName">
  <param
        paramName="paramName"
        paramDefaultValue= "thevalue">
    <!--in case the value is a list-->
    <values>
      <value key="value" value="value"/>
    </values>
  </param>
</XDEPInlineResponse>
```

Перший запит, що відправляється до серверної команди має бути ProtocolSyntax, для того, щоб отримати опис всіх запитів та відповідей доступних в конкретній серверній реалізації протоколу, та порядок їх виконання.

```
<ProtocolSyntax>
  <parameters>
    <param paramName="devicename"
paramValue="the_name_of_the_device"/>
  </parameters>
</ ProtocolSyntax >
```

Так як протокол вимагає виконання команди ProtocolSyntax, вона має бути закодована в клієнстську програму. Відповідь на цей запит надає користувачу синтаксис протоколу. Такий підхід дозволяє в універсальній формі

1. Конструювати інтерфейси користувача не прив'язуючись до конкретного візуального представлення даних;
2. Визначати кроки виконання програми.

Ідея цього підходу нагадує принципи роботи HTTP. Однак, існує три суттєві особливості, що відрізняють XDEP від HTTP [6] та веб-сервісів [2], [3]:

- Команди добре структуровані, що дозволяє їх ефективно оброблювати машиною;
- Інтерфейс користувача конструюється під час роботи програми;
- Чітко визначені кроки виконання та порядку обміну командами.

Третя особливість вимагає додаткового розгляду. Ідентифікація порядку виконання команд досягається через теги startcommand, successors та predecessorParamReference (див. синтаксис протоколу) . Перший тег використовується для визначення першої команди циклу виконання програми. Цю команду клієнт-програма має відправити першою на сервер. Це є початкова точка виконання програми. Наступний тег successors надає перелік команд, які можуть бути виконані після поточної команди. Важливо зауважити, що презентація цієї інформації для користувача повністю залежить від клієнт-програми. Тег successor містить autocall параметер, який визначає - чи виконується наступна команда після закінчення виконання поточної команди автоматично. Така функціональність необхідна в разі, коли програма має виконувати деякий послідовний набір команд. Останній тег predecessorParamReference використовується для визначення параметрів попередньої команди, дані з яких треба передати в поточний параметр. Таке представлення допомогає передати дані з однієї команди до іншої, в разі, якщо ці дані надходять з сервера. Ця особливість ставить XDEP окремо від протоколів, які не підтримують поняття стану[4]. Вона досягається через перенесення логіки послідовності виконання програми на клієнт-частину. Цей підхід виражає поняття сесії, який навіть сьогодні не підтримується в веб-сервісах.

Для того, щоб чітко зрозуміти зміст використання цієї функціональності розглянемо ще раз проект Fenestra. У зв'язку з тим, що клієнт програма працює в аудіо режимі, після виконання кожної з команд користувач від персональної консолі отримує голосове повідомлення про результат виконання запиту та переліком можливих наступних команд. Користувач голосом обирав команду для виконання. Як тільки пристрій успішно розпізнавав голосову команду, користувачу пропонувалось вказати дані для параметрів запиту. В разі, якщо параметри вимагали визначених даних, їх перелік зачитувався користувачеві. Коли всі голосові повідомлення інтерпретувались однозначно, клієнт-програма відправляла команду на сервер і процес починався спочатку.

Табл. 1 містить перелік всіх тегів та коментарі до них, які використовуються в цій програмі.

Кожна команда вимагає введення різних параметрів цілого, строкового, або інших типів. Додатково, деякі параметри можуть мати не одне значення, а їх набір.

Таблиця 1.

Перелік тегів

Назва тегу	Опис
XDEPDriver	Базовий тег для опису протоколу
ProtocolName	І'мя протоколу
ServerName	І'мя серверу
startcommand	Стартова команда
Command	Мета опис конкретної команди
Name	І'мя команди
IsStatic	Визначає чи доступна команда в будь-який момент часу
Description	Опис команди
Parameters	Мета опис параметрів
ParamName	І'мя параметру
predecessorParamReference	Посилання на параметр з минулої команди
paramType	Тип параметру *integer|boolean|string|list;*
paramDefaultValue	Значення по-замовчуванню
maxValues	Максимальна довжина значення параметру
minValues	Мінімальна довжина значення параметру
validationRule	Правило верифікації значення визначене через регулярні відносини
Direction	Визначає як оброблюється параметр. Можливі варіанти: in|out|inout
Values	Тег з переліком передвизначених значень параметру
Key	Ключ переліку
Value	Значення переліку
Successors	Перелік команд, які доступні користувачеві після закінчення обробки поточної команди
CommandName	І'мя команди
Autocall	Чи автоматично визивається команда

Під час розгляду представленого підходу, автор зробив акцент на перевагах протоколу з точки зору користувачів. Однак, структурність протоколу також мотивує його технологічне застосування. Згідно агенції досліджень Juniper Research, кількість користувачів, які будуть активно використовувати їх мобільні пристрої до 2011 складе 54 мілйони. А кількість їх транзакцій складатиме 11,5 $ міліардів [5]. Це відкриває нові простори для застосування протоколу.

Зараз, розглянемо ще один раз приклад із велосипедними станціями з перспективи розробки протоколу на базі XDEP. Можна сміливо заявити, що 100% користувачів кредитних карток володіють мобільним телефоном. Припустимо, що замість автоматичних пунктів, керівництво міста облаштувала платформи Bluetooth® серверами.

Актор (мандрівник) наближається до пункту з орендою велосипедів. На його мобільному телефоні встановлена програма які підтримує функції протоколів XDEP. Ця програма автоматично фіксує серверну платформу через Bluetooth® канали. В разі, якщо користувач вперше користується послугою, програма вимагає ввести деталі кредитної картки (можливо, якщо користувач повністю довіряє інформаційним системам, він вкаже номер кредитної картки, який програма зможе використовувати всюди), в іншому випадку, програма автоматично відправляє дані про взятий у поточний момент велосипед (час, номер та платформа оренди). Таким чином актор може продовжити свою подорож не змінюючи велосипед та не втрачаючи час на формальні процедури. До того ж, такий підхід дозволить знизити вартість обладнання нових платформ та їх майбутнє обслуговування. В разі ж, якщо користувач не має відповідну програму на своєму пристрої, вона може бути завантажена через ті ж бездротові канали.

Висновки. В даній статті було розглянуто еволюцію класу протоколів XDEP, який призвів до нової можливості конструювання інтерфейсів користувачів незалежно від типу та функціонального призначення мобільних приладів. Основною ціллю цього рішення була автоматизація процесів, необхідних для надання різноманітних послуг для різних груп користувачів. Ця мета була досягнена через розробку класу платформонезалежних протоколів. У роботі представлено два приклади, один з яких - програмний проект Fenestra розроблений для конкурсу Microsoft Imagine Cup. Другий, – розглянутий в перспективі застосування для вирішення недосконалостей існуючої системи, Velolib. В наступних роботах автор продовжить розвинення XDEP через більш детальне дослідження проблем верифікації даних та методів обробки помилок. Як тількі ци питання будуть остаточно вирішені, система буде готова до запуску в промислових проектах.

Список літератури: 1. Doroshenko, A; Yatsenko, K: Protocols for Mobile Devices Integration in Heterogeneous Environments // Proc. ISTA'2007. **2.** Graham, S; Davis, D. et al. Building Web Services with Java : Making Sense of XML, SOAP, WSDL, and UDDI (2nd Edition) (Developer's Library), 2004. **3.** Rogerson, D.: Inside Com (Microsoft Programming Series), 1997. **4.** Troelsen, A.: COM and .NET Interoperability, 2002. **5.** http://zoom.cnews.ru/blog/?p=5847, 2007. **6.** Gundavaram, S.: CGI Programming on the World Wide Web (Nutshell Handbook), 1996. **7.** Swanke, J: COM Programming by Example: Using MFC, ActiveX, ATL, ADO, and COM+ (with CD-ROM), 2000. **8.** Foggon ,D; Maharry D.; Ullman, C.; Watson, K.; Programming Microsoft .NET XML Web Services (Pro-Developer), 2003. **9.** Куроуз Дж., Росс К., Кузнецов А.; Компьютерные сети: Многоуровневая архитектура Интернета, 2004. **10.** Заика А.; Компьютерные сети, 2006. **11.** Наконечний В., Тарасов Д.; Інтелектуальна інформаційна система моніторингу та аналізу стану безпеки в мережі інтернет, 2007. **12.**Тарасов Д., Сєров Ю.; Методи побудови тематичних каталогів тегів на основі сервісів закладок.

Поступила в редколегію 24.02.2008